Tightly coupling Speech Recognition and Search

Taniya Mishra
AT&T Labs-Research
180 Park Ave
Florham Park, NJ 07932
taniya@research.att.com

Srinivas Bangalore
AT&T Labs-Research
180 Park Ave
Florham Park, NJ 07932
srini@research.att.com

Abstract

In this paper, we discuss the benefits of tightly coupling speech recognition and search components in the context of a speech-driven search application. We demonstrate that by incorporating constraints from the information repository that is being searched not only improves the speech recognition accuracy but also results in higher search accuracy.

1 Introduction

With the exponential growth in the use of mobile devices in recent years, the need for speech-driven interfaces is becoming apparent. The limited screen space and soft keyboards of mobile devices make it cumbersome to type in text input. Furthermore, by the *mobile* nature of these devices, users often would like to use them in hands-busy environments, ruling out the possibility of typing text.

In this paper, we focus on the problem of speech-driven search to access information repositories using mobile devices. Such an application typically uses a speech recognizer (ASR) for transforming the user's speech input to text and a search component that uses the resulting text as a query to retrieve the relevant documents from the information repository. For the purposes of this paper, we use the business listings containing the name, address and phone number of businesses as the information repository.

Most of the literature on speech-driven search applications that are available in the consumer market (Acero et al., 2008; Bacchiani et al., 2008; VLingo FIND, 2009) have quite rightly emphasized the importance of the robustness of the ASR language model and the data needed to build such a robust language model. We acknowledge that this is a significant issue for building such systems, and we provide our approach to creating a language model. However, in contrast to most of these systems that treat speech-driven search to be largely an ASR problem followed by a Search problem, in this paper, we show the benefits of tightly coupling ASR and Search tasks and illustrate techniques to improve the accuracy of both components by exploiting the co-constraints between the two components.

The outline of the paper is as follows. In Section 2, we discuss the set up of our speech-driven application. In Section 3, we discuss our method to integrating the speech and search components. We present the results of the experiments in Section 4 and conclude in Section 5.

2 Speech-driven Search

We describe the speech-driven search application in this section. The user of this application provides a speech utterance to a mobile device intending to search for the address and phone number of a business. The speech utterance typically contains a business name, optionally followed by a city and state to indicate the location of the business (e.g. *pizza hut near urbana illinois.*). User input with a business category (*laundromats in madison*) and without location information (*hospitals*) are some variants supported by this application. The result of ASR is used to search a business listing database of over 10 million entries to retrieve the entries pertinent to the user query.

The ASR used to recognize these utterances incorporates an acoustic model adapted to speech collected from mobile devices and a trigram language model that is built from over 10 million text query logs obtained from the web-based text-driven version of this application. The 1-best speech recognition output is used to retrieve the relevant business listing entries.

3 Tightly coupling ASR and Search

As mentioned earlier, most of the speech-driven search systems use the the 1-best output from the ASR as the query for the search component. Given that ASR 1-best output is likely to be erroneous, this serialization of the ASR and search components might result in sub-optimal search accuracy. As will be shown in our experiments, the oracle word/phrase accuracy using n-best hypotheses is far greater than the 1-best output. However, using each of the n-best hypothesis as a query to the search component is computationally sub-optimal since the strings in the n-best hypotheses usually share large subsequences with each other. A lattice representation of the ASR output, in particular, a word-confusion network (WCN) transformation of the lattice, compactly encodes the n-best hypothesis with the flexibility of pruning alternatives at each word position. An example of a WCN is shown in Figure 1. In order to obtain a measure of the ambiguity per word position in the WCN, we define the (average) *arc density* of a WCN as the ratio of the total number of arcs to the number of states in the WCN. As can be seen, with very small increase in arc density, the number of paths that are encoded in the WCN can be increased exponentially. In Figure 2, we show the improvement in oracle-path word and phrase accuracies as a function of the arc density for our data set. Oracle-path is a path in the WCN that has the least edit-distance (Levenshtein, 1966) to the reference string. It is interesting to note that the oracle accuracies can be improved by almost 10% absolute over the 1-best accuracy with small increase in the arc density.

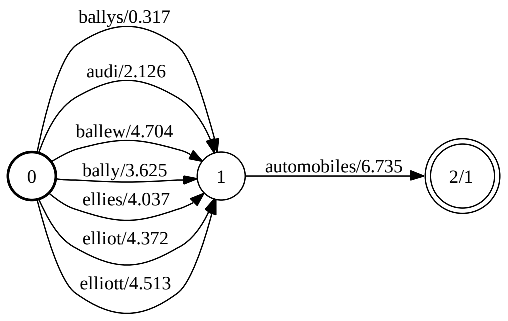

Figure 1: A sample word confusion network

3.1 Representing Search Index as an FST

In order to exploit WCNs for Search, we have implemented our own search engine instead of using an

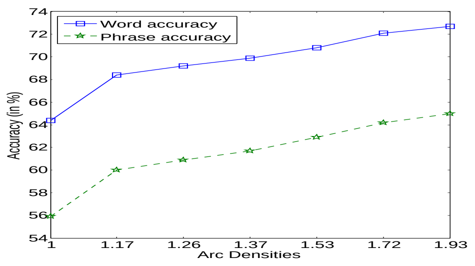

Figure 2: Oracle accuracy graph for the WCNs at different arc densities

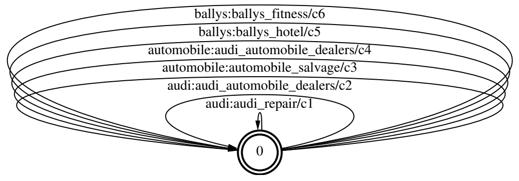
ballys:ballys_fitness/c6
ballys:ballys_hotel/c5
automobile:audi_automobile_dealers/c4
automobile:automobile_salvage/c3
audi:audi_automobile_dealers/c2
audi:audi_repair/c1
0

Figure 3: An example of an FST representing the search index

off-the-shelf search engine such as Lucene (Hatcher and Gospodnetic., 2004). We index each business listing (d) in our data that we intend to search using the words (w_d) in that listing. The pair (w_d, d) is assigned a weight ($c_{(w_d,d)}$) using different metrics, including the standard $tf * idf$, as explained below. This index is represented as a weighted finite-state transducer (*SearchFST*) as shown in Figure 3 where w_d is the input symbol, d is the output symbol and $c_{(w_d,d)}$ is the weight of that arc.

3.2 Relevance Metrics

In this section, we describe six different weighting metrics used to determine the relevance of a document for a given query word that we have experimented with in this paper.

idf_w: idf_w refers to the inverse document frequency of the word, w, which is computed as $ln(D/d_w)$, where D refers to the total number of documents in the collection, and d_w refers to the total number of documents in the collection that contain the word, w (Robertson and Jones, 1997; Robertson, 2004).

atf_w: atf_w refers to average term frequency, which is computed as cf_w/d_w (Pirkola et al., 2002).

$cf_w \times idf_w$: Here cf_w refers to the collection frequency, which is simply the total number of occurrences of the word, w in the collection.

$atf_w \times idf_w$: (Each term as described above).

$\sum \frac{f_{w,d}}{|d_w|} \times idf_w$: Here $f_{w,d}$ refers to the frequency of the word, w, in the document, d, whereas $|d_w|$ is the length of the document, d, in which the word, w, occurs.

$\frac{cf_w}{\sum |d_w|} \times idf_w$: (Each term as described above).

3.3 Search

By composing a query (*Qfst*) (either a 1-best string represented as a finite-state acceptor, or a WCN), with the *SearchFST*, we obtain all the arcs $(w_q, d_{w_q}, c_{(w_q, d_{w_q})})$ where w_q is a query word, d_{w_q} is a listing with the query word and, $c_{(w_q, d_{w_q})}$ is the weight associated with that pair. Using this information, we aggregate the weight for a listing (d_q) across all query words and rank the retrieved listings in the descending order of this aggregated weight. We select the top N listings from this ranked list. The query composition, listing weight aggregation and selection of top N listings are computed with finite-state transducer operations.

In Figure 4, we illustrate the result of reranking the WCN shown in Figure 1 using the search relevance weights of each word in the WCN. It must be noted that the least cost path[1] for the WCN in Figure 1 is *ballys automobiles* while the reranked 1-best output in Figure 4 is *audi automobiles*. Given that the user voice query was *audi automobiles*, the listings retrieved from the 1-best output after reranking are much more relevant than those retrieved before reranking, as shown in Table 1.

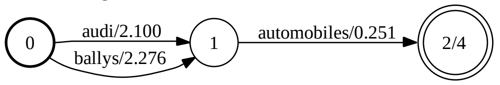

Figure 4: A WCN rescored using word-level search relevance weights.

4 Experiments and Results

We took 852 speech queries collected from users using a mobile device based speech search application. We ran the speech recognizer on these queries using the language model described in Section 2 and created word-confusion networks such as those illustrated in Figure 1. These 852 utterances were divided into 300 utterances for the development set and 552 for the test set.

[1]We transform the scores into costs and search for minimum cost paths.

Before rescoring	After rescoring
ballys intl los angeles ca	auburn audi repair auburn wa
ballys las vegas las vegas nv	audi bellevue repair bellevue wa
ballys las health spa las vegas nv	university audi seattle wa
ballys cleaners palm desert ca	beverly hills audi los angeles ca
ballys brothers yorba linda ca	audi independent repairs by eurotech livermore ca

Table 1: Listings retrieved for query *audi automobiles* before and after ASR WCNs were rescored using search relevance weights.

4.1 ASR Experiments

The baseline ASR word and sentence (complete string) accuracies on the development set are 63.1% and 57.0% while those on the test set are 65.1% and 55.3% respectively.

Metric	Word Acc.	Sent. Acc.	Scaling Factor	AD
idf_w	63.1	57.0	$10^{-}3$	all
$cf_w \times idf_w$	63.5	58.3	$15 * 10^{-}4$	1.37
atf_w	63.6	57.3	1	all
$atf_w \times idf$	63.1	57.0	$10^{-}3$	all
$\sum \frac{f_{w,d}}{\lvert df_w \rvert} \times idf$	63.9	58.3	$15 * 10^{-}4$	1.25
$\frac{cf_w}{\sum \lvert df_w \rvert} \times idf_w$	63.5	57.3	1	all

Table 2: Performance of the metrics used for rescoring the WCNs output by ASR. (AD refers to *arc density*.)

In Table 2, we summarize the improvements obtained by rescoring the ASR WCNs based on the different metrics used for computing the word scores according to the search criteria. The largest improvement in word and sentence accuracies is obtained by using the rescoring metric: $\sum \frac{f_{w,d}}{|df_w|} \times idf$. The word-level accuracy improved from the baseline accuracy of 63.1% to 63.9% after rescoring while the sentence-level accuracy improved from 57.0% to 58.3%. Thus, this rescoring metric, and the corresponding pruning AD and the scaling factor was used to rerank the 552 WCNs in the test set. After rescoring, on the test set, the word-level accuracy improved from 65.1% to 65.9% and sentence-level accuracy improved from 55.3% to 56.2%.

Number of documents	Scores	Baseline	Reranked
	Precision	0.708	0.728
All	Recall	0.728	0.742
Documents	F-Score	0.718	0.735

Table 3: Table showing the relevancy of the search results obtained by the baseline ASR output compared to those obtained by the reranked ASR output.

4.2 Search Experiments

To analyze the Search accuracy of the baseline ASR output in comparison to the ASR output, reranked using the $\sum \frac{f_{w,d}}{|df_w|} \times idf$ reranking metric, we used each of the two sets of ASR outputs (i.e., baseline and reranked) as queries to our search engine, *SearchFST* (described in Section 3). For the search results produced by each set of queries, we computed the precision, recall, and F-score values of the listings retrieved with respect to the listings retrieved by the set of human transcribed queries (*Reference*). The precision, recall, and F-scores for the baseline ASR output and the reranked ASR output, averaged across each set, is presented in Table 3. For the purposes of this experiment, we assume that the set returned by our *SearchFST* for the human transcribed set of queries is the reference search set. This is however an approximation for a human annotated search set.

In Table 3, by comparing the search accuracy scores corresponding to the baseline ASR output to those corresponding to the reranked ASR output, we see that reranking the ASR output using the information repository produces a substantial improvement in the accuracy of the search results.

It is interesting to note that even though the reranking of the ASR as shown in Table 2 is of the order of 1%, the improvement in Search accuracy is substantially higher. This indicates to the fact that exploiting constraints from both components results in improving the recognition accuracy of that subset of words that are more relevant for Search.

5 Conclusion

In this paper, we have presented techniques for tightly coupling ASR and Search. The central idea behind these techniques is to rerank the ASR output using the constraints (encoded as relevance metrics) from the Search task. The relevance metric that best improved accuracy is $\sum \frac{f_{w,d}}{|d_w|} \times idf_w$, as determined on our development set. Using this metric to rerank the ASR output of our test set, we improved ASR accuracy from 65.1% to 65.9% at the word-level and from 55.3% to 56.2% at the phrase level. This reranking also improved the F-score of the search component from 0.718 to 0.735. These results bear out our expectation that tightly coupling ASR and Search can improve the accuracy of both components.

Encouraged by the results of our experiments, we plan to explore other relevance metrics that can encode more sophisticated constraints such as the relative coherence of the terms within a query.

Acknowledgments

The data used in this work is partly derived from the Speak4It voice search prototype. We wish to thank every member of that team for having deployed that voice search system.

References

A. Acero, N. Bernstein, R.Chambers, Y. Ju, X. Li, J. Odell, O. Scholtz P. Nguyen, and G. Zweig. 2008. Live search for mobile: Web services by voice on the cellphone. In *Proceedings of ICASSP 2008*, Las Vegas.

M. Bacchiani, F. Beaufays, J. Schalkwyk, M. Schuster, and B. Strope. 2008. Deploying GOOG-411: Early lesstons in data, measurement and testing. In *Proceedings of ICASSP 2008*, Las Vegas.

E. Hatcher and O. Gospodnetic. 2004. *Lucene in Action (In Action series)*. Manning Publications Co., Greenwich, CT, USA.

V.I. Levenshtein. 1966. Binary codes capable of correcting deletions, insertion and reversals. *Soviet Physics Doklady*, 10:707–710.

A. Pirkola, E. Lepaänen, and K. Järvelin. 2002. The ”ratf” formula (kwok's formula): exploiting average term frequency in cross-language retrieval. *Information Research*, 7(2).

S. E. Robertson and K. Sparck Jones. 1997. Simple proven approaches to text retrieval. Technical report, Cambridge University.

Stephen Robertson. 2004. Understanding inverse document frequency: On theoretical arguments for idf. *Journal of Documentation*, 60.

VLingo FIND, 2009. *http://www.vlingomobile.com/downloads.html*.